

Quantum Query as a State Decomposition

S. A. Grillo¹ and F. L. Marquezino^{1,2}

¹PESC/Coppe and ²Numpex-Comp/Xerem
Universidade Federal do Rio de Janeiro
Rio de Janeiro, Brazil

{sgrillo, franklin}@cos.ufrj.br

November 30, 2016

Abstract

The Quantum Query Model is a framework that allows us to express most known quantum algorithms. Algorithms represented by this model consist on a set of unitary operators acting over a finite Hilbert space, and a final measurement step consisting on a set of projectors. In this work, we prove that the application of these unitary operators before the measurement step is equivalent to decomposing a unit vector into a sum of vectors and then inverting some of their relative phases. We also prove that the vectors of that sum must fulfill a list of properties and we call such vectors a *Block Set*. If we define the measurement step for the Block Set Formulation similarly to the Quantum Query Model, then we prove that both formulations give the same Gram matrix of output states, although the Block Set Formulation allows a much more explicit form. Therefore, the Block Set reformulation of the Quantum Query Model gives us an alternative interpretation on how quantum algorithms works. Finally, we apply our approach to the analysis and complexity of quantum exact algorithms.

Keywords: quantum exact algorithms, quantum query complexity, computational complexity, analysis of algorithms, design of algorithms.

1 Introduction

The Quantum Query Model (QQM) is an important tool in the analysis and design of quantum algorithms, especially because its simplicity allows us to compare classical and quantum computing more easily. This model generalizes decision trees [10] with complexity being defined as the minimum number of oracle queries required for computing a given function f for any input $x \in \{0, 1\}^n$.

The topic of exact quantum algorithms is less understood than bounded-error algorithms. For many years, the only exact quantum algorithms known to produce a speed-up over classical algorithms for total functions were those that used Deutsch’s algorithm as a subroutine [3]. The numerical method proposed by Barnum *et al.* [8] just gives us approximate solutions, whose results can be laborious to translate into analytically defined algorithms for the exact case [17]. Currently, there is a limited number of research papers that presents results in the analytic construction of exact quantum algorithms [3, 17, 7, 4, 13, 6].

In query complexity, the polynomial method [9] and adversary methods [2, 14] are well known for computing lower-bounds of exact quantum algorithms. There are important results about exact quantum query complexity in the literature obtained from such methods:

- For an exact quantum algorithm that computes a total Boolean function within t queries, there is a classical deterministic algorithm that computes the same function by applying $\mathcal{O}(t^3)$ queries [16].
- Exact quantum algorithms give an advantage for all Boolean functions excepting AND_n [5].

In our present work, we propose another reformulation of the QQM and we apply it to the analysis and complexity of exact quantum algorithms. The new model proposed in this paper, called *Block Set Formulation* (BSF), is shown to be equivalent to the QQM. In this formulation, the algorithm is represented by a set of vectors satisfying certain properties, and the unitary operators are replaced by phase inversions on some of those vectors. This set of vectors, called *Block Set*, gives an alternative interpretation on how quantum algorithms work. For each input, the BSF constructs a corresponding output state following a different definition to the QQM. After applying the measurement step on such output state, however, the results are identical in both models. The equivalence between BSF and QQM is proved on two steps. First, we prove that for each QQM algorithm of t queries there is a unique t -dimensional Block Set with the same Gram matrix of output states. Then, we prove that for each t -dimensional Block Set there are several QQM algorithms of t queries with the same Gram matrix of output states. Considering that two algorithms with the same Gram matrix of output states are similar—since we can choose the measurement operators appropriately—then the QQM and the BSF are equivalent.

The BSF can be simplified by proving that a BSF restricted to real numbers is equivalent to a complex BSF. Assuming a real-valued BSF, we prove that

the Gram matrix of output states is equal to a sum of matrices, where each of them depends on some pair of elements in the Block Set. By using the relation between Block Sets and the final Gram matrix we obtain a necessary and sufficient condition for the existence of exact quantum algorithms, this condition being formulated by a system of equations that requires a semi-definite solution. If we consider a special case of Block Sets in which all elements are pairwise orthogonal, we obtain a second condition for exact quantum algorithms from our first system. This second condition is just sufficient for exact quantum algorithms; nonetheless, we show that it can also be used as an analytic tool for constructing exact quantum algorithms. As an example of the application of orthogonal Block Sets, we define the *XOR-Weighted-Problem* and prove that it can be solved by exact quantum algorithms in the BSF, in which case we also give an upper-bound for its complexity. Finally, we present a lower-bound for the exact quantum query complexity of functions with Boolean domain and arbitrary output.

The structure of our paper is as follows. In Sec. 2, we briefly review the basic concepts and formulations of the Quantum Query Model. In Sec. 3, we introduce the Block Set Formulation and prove its equivalence to the Quantum Query Model. In Sec. 4, we present the relation between Block Sets and the Gram matrix of output states. In Sec. 5, we show a linear condition for quantum exact algorithms and obtain a model that characterizes the computing power of a family of QQM algorithms. In Sec. 6, we prove a lower-bound for exact quantum query algorithms. In Sec. 7, we present our conclusion and discuss potential extensions of this approach. At last, in the appendices, we present examples.

2 Preliminaries

Let H be a finite Hilbert space and let T be a finite set. Two operators A and B are orthogonal if $\langle \Psi | A^\dagger B | \Phi \rangle = 0$ for all $|\Phi\rangle, |\Psi\rangle \in H$. A Complete Set of Orthogonal Projectors (CSOP) is an indexed set of pairwise orthogonal projectors $\{P_z : z \in T\}$, satisfying

$$\sum_{z \in T} P_z = I_H, \tag{1}$$

where I_H is the identity operator on H . We denote H_z^P and H_z^Q , as the ranges of spaces projected by P_z and Q_z respectively.

Lemma 1. *If $\{P_z : z \in T\}$ is a CSOP and U is a unitary operator, then*

$$\{U^\dagger P_z U : z \in T\}$$

is a CSOP.

Lemma 2. *Let $\{P_z : z \in T\}$ and $\{Q_z : z \in T\}$ be two CSOP over H such that $\dim(H_z^P) = \dim(H_z^Q)$ for the same z . Then, there exists a unitary operator U such that $U^\dagger P_z U = Q_z$ for all $z \in T$.*

The Quantum Query Model (QQM) is a formulation that simplifies the analysis of quantum algorithms for computing a function $f : \{0, 1\}^n \rightarrow \{0, 1\}$ for an input x making *queries* to the values x_i . In this model, we are mostly concerned with the number of queries to the input. In the QQM, the memory can be divided in two registers: (i) the *query register*, whose size should allow it to represent any integer $i \in \{0, \dots, n\}$, for an input of size n ; and (ii) the *working memory*, without size constraints.

The query register and the working memory are jointly known as the *accessible memory*. The computational basis for the associated Hilbert space H_A (or, *accessible space*) is composed by the vectors $|i, w\rangle$ where $i \in \{0, \dots, n\}$ and w is a possible state of the working memory. Thus, we can define the Hilbert spaces associated to each register: (i) the *query space* H_Q is spanned by vectors $\{|i\rangle : 0 \leq i \leq n\}$; (ii) the *work space* H_W is spanned by vectors $|w\rangle$, where w is in the set of allowed values for the working memory. Hence, $H_A = H_Q \otimes H_W$. If $|\Psi\rangle \in H_A$, then it can be written uniquely as

$$|\Psi\rangle = \sum_{i=0}^n |i\rangle |\Psi_i\rangle, \quad (2)$$

where $|\Psi_i\rangle \in H_W$. The oracle operator O_x for some input $x \in \{0, 1\}^n$ is defined as

$$O_x |i\rangle |\Psi_i\rangle = (-1)^{x_i} |i\rangle |\Psi_i\rangle, \quad (3)$$

where query space has dimension $n+1$ and $x_0 = 0$ is not considered part of the input. In our setting it is very important to define $x_0 = 0$, otherwise we could not compute a wide range of functions. However, it can be avoided in other equivalent descriptions of QQM, by using a slightly different definition of the

oracle operator [15].

A quantum query algorithm with an output domain T is determined by: (i) the number b of qubits in the working memory; (ii) a sequence of unitary operators $\{U_i : 0 \leq i \leq t\}$ in H_A ; and (iii) a CSOP over H_A with projectors indexed by elements of T for the final measurement.

The execution of the algorithm for input x produces a final state

$$|\Psi_x^f\rangle = U_t O_x U_{t-1} \dots U_1 O_x U_0 |0, 0\rangle. \quad (4)$$

The number of queries is defined as the number of times O_x occurs in the execution. The output $z \in T$ is chosen with a probability $\pi_x(z) = \|P_z |\Psi_x^f\rangle\|^2$, using the CSOP.

We say that an algorithm computes a function $f : \{0, 1\}^n \rightarrow T$ within error ε if for all input x , there is $\pi_x(f(x)) \geq 1 - \varepsilon$. An algorithm is *exact* if $\varepsilon = 0$ and is *bounded-error* if $\varepsilon \leq 1/3$.

3 A reformulation of the Quantum Query Model

First, we need to introduce a sequence of unitary operators

$$\tilde{U}_k = U_k U_{k-1} \dots U_0$$

for $t \geq k$. Let $\{P_k : 0 \leq k \leq n\}$ be a CSOP where each P_i has as range the subspace of vectors of the form $|i\rangle |\psi\rangle$, where $|i\rangle \in H_Q$ and $|\psi\rangle \in H_W$. Finally, we introduce the notation $\tilde{P}_i^j = \tilde{U}_j^\dagger P_i \tilde{U}_j$. From Lemma 1 we know that $\{\tilde{P}_k^j : 0 \leq k \leq n\}$ is also a CSOP for any fixed j .

We denote an *algorithm* (without the measurement step) by the 7-tuple

$$\mathcal{A} = (t, n, m, H_Q, H_W, \Psi, \{U_i\}),$$

where $\dim(H_Q) = n + 1$, $\dim(H_W) = m$ and $|\Psi\rangle \in H_A$ is a unit vector and the unitary operators in $\{U_i : 0 \leq i \leq t + 1\}$ are defined on H_A . In the present work, we always consider algorithms according to the above definition, unless otherwise stated.

This is all the information required for describing an algorithm using $t + 1$ queries and initial state $|\Psi\rangle$. Choosing an arbitrary initial state $|\Psi\rangle$ is the same as using $|0, 0\rangle$, but we change the convention because is more convenient for

upcoming notations.

Definition 1. Let $a = (a_0, a_1, \dots, a_t)$ be a vector and let $\mathbb{Z}_{n+1} = \{0, 1, \dots, n\}$. We say that an indexed set of vectors $\{|\Psi(k)\rangle \in H_A : k \in \mathbb{Z}_{n+1}^{t+1}\}$ is associated with

$$\mathcal{A} = (t, n, m, H_Q, H_W, \Psi, \{U_i\})$$

if we have that

$$|\Psi(a)\rangle = \tilde{P}_{a_t}^t \dots \tilde{P}_{a_1}^1 \tilde{P}_{a_0}^0 |\Psi\rangle, \quad (5)$$

where $0 \leq a_i \leq n$ for all i .

The motivation for this definition is better understood by considering the following equation. Notice that, by using the operators defined above, we have the expression

$$\begin{aligned} |\Psi\rangle &= \left(\sum_{k_t=0}^n \tilde{P}_{k_t}^t \right) \dots \left(\sum_{k_0=0}^n \tilde{P}_{k_0}^0 \right) |\Psi\rangle \\ &= \sum_{k_t=0}^n \dots \sum_{k_0=0}^n |\Psi(k_0, \dots, k_t)\rangle. \end{aligned} \quad (6)$$

for any vector. If $|\Psi\rangle$ is a unit vector then the set $\{|\Psi(k_0, \dots, k_t)\rangle\}$ is associated with some algorithm \mathcal{A} , whose initial state is $|\Psi\rangle$ and has $t+1$ queries. Thus, we can interpret Eq. (6) as a decomposition of an initial state.

This decomposition has an important property that will be given by Theorem 1. However, we need to introduce a useful identity first. Since

$$\{\tilde{P}_i^j : 0 \leq i \leq n\}$$

is a CSOP for each j and

$$O_x |\Psi\rangle = \sum_{i \in \{k: x_k=0\}} P_i |\Psi\rangle - \sum_{i \in \{k: x_k=1\}} P_i |\Psi\rangle, \quad (7)$$

we get

$$\tilde{U}_j^\dagger O_x \tilde{U}_j |\Psi\rangle = \sum_{i \in \{k: x_k=0\}} \tilde{U}_j^\dagger P_i \tilde{U}_j |\Psi\rangle - \sum_{i \in \{k: x_k=1\}} \tilde{U}_j^\dagger P_i \tilde{U}_j |\Psi\rangle. \quad (8)$$

Theorem 1. If the indexed vector $\{|\Psi(k)\rangle \in H_A : k \in \mathbb{Z}_{n+1}^{t+1}\}$ is associated with

the algorithm $\mathcal{A} = (t, n, m, H_Q, H_W, \Psi, \{U_i\})$, then

$$\tilde{U}_t^\dagger O_x U_t \dots U_1 O_x U_0 |\Psi\rangle = \sum_{k_t=0}^n \dots \sum_{k_0=0}^n (-1)^{\sum_{i=0}^t x_{k_i}} |\Psi(k_0, \dots, k_t)\rangle. \quad (9)$$

Proof. We shall prove by induction on t . First, as we state our induction hypothesis, notice that Eq. (9) holds when $t = 0$:

$$\begin{aligned} \tilde{U}_0^\dagger O_x U_0 |\Psi\rangle &= U_0^\dagger O_x U_0 |\Psi\rangle \\ &= \sum_{k_0=0}^n (-1)^{x_{k_0}} |\Psi(k_0)\rangle. \end{aligned} \quad (10)$$

Then, notice that if Eq. (9) holds for a particular t , then it must hold for $t + 1$. That is, if

$$\tilde{U}_t^\dagger O_x \dots O_x U_0 |\Psi\rangle = \sum_{k_t=0}^n \dots \sum_{k_0=0}^n (-1)^{\sum_{i=0}^t x_{k_i}} |\Psi(k_0, \dots, k_t)\rangle$$

then, using the Eq. (8), we have that

$$\tilde{U}_{t+1}^\dagger O_x \dots O_x U_0 |\Psi\rangle = \sum_{k_t=0}^n \dots \sum_{k_0=0}^n (-1)^{\sum_{i=0}^t x_{k_i}} \sum_{k_{t+1}=0}^n (-1)^{x_{k_{t+1}}} \tilde{P}_{k_{t+1}}^{t+1} |\Psi(k_0, \dots, k_t)\rangle.$$

Reordering the summations, we have

$$\tilde{U}_{t+1}^\dagger O_x \dots O_x U_0 |\Psi\rangle = \sum_{k_{t+1}=0}^n \dots \sum_{k_0=0}^n (-1)^{\sum_{i=0}^{t+1} x_{k_i}} \tilde{P}_{k_{t+1}}^{t+1} |\Psi(k_0, \dots, k_t)\rangle.$$

According to Definition 1 and observing the notation for \tilde{P}_i^j , we finally have

$$\tilde{U}_{t+1}^\dagger O_x \dots O_x U_0 |\Psi\rangle = \sum_{k_{t+1}=0}^n \dots \sum_{k_0=0}^n (-1)^{\sum_{i=0}^{t+1} x_{k_i}} |\Psi(k_0, \dots, k_{t+1})\rangle.$$

□

Corollary 1. Consider the vectors $|\bar{\Psi}(k_0, \dots, k_t)\rangle = U_{t+1} \tilde{U}_t |\Psi(k_0, \dots, k_t)\rangle$ $\forall k_i \in \mathbb{Z}_{n+1}$. If $\{|\Psi(k)\rangle \in H_A : k \in \mathbb{Z}_{n+1}^{t+1}\}$ is associated with

$$\mathcal{A} = (t, n, m, H_Q, H_W, \Psi, \{U_i\})$$

then

$$U_{t+1}O_xU_t\ldots U_1O_xU_0|\Psi\rangle = \sum_{k_t=0}^n \ldots \sum_{k_0=0}^n (-1)^{\sum_{i=0}^t x_{k_i}} |\bar{\Psi}(k_0, \dots, k_t)\rangle. \quad (11)$$

Corollary 1 shows that any quantum algorithm can be represented as a sum of invariant vectors, whose signs are changed depending on the input. In Fig. 1, we can see an example by means of a graphical representation.

Notice that the algorithm of Corollary 1 is equivalent to the algorithm of Theorem 1, because both algorithms have the same Gram matrices for the final states $|\Psi_x^f\rangle$. So we can ignore the last unitary operator U_{t+1} and conclude that the algorithm is determined by the vectors that appear in the decomposition, see Eq. (6). The following definition and theorems show us that we can reformulate the QQM by using this decomposition instead of unitary operators.

Definition 2 (Block Set). *Let $n, t \geq 0$. We say that an indexed set $\{|\Psi(k)\rangle \in H_1 \otimes H_2 : k \in \mathbb{Z}_{n+1}^{t+1}\}$ is a Block Set for the ordered pair of Hilbert spaces (H_1, H_2) , if:*

- $\langle \Psi^i(b_0, \dots, b_{t-i}) | \Psi^i(c_0, \dots, c_{t-i}) \rangle = 0$ if $b_{t-i} \neq c_{t-i}$ for $0 \leq i \leq t$, where the vector $|\Psi^i(a_0, \dots, a_{t-i})\rangle$ is defined as

$$\sum_{k_1=0}^n \ldots \sum_{k_i=0}^n |\Psi(a_0, \dots, a_{t-i}, k_1, \dots, k_i)\rangle;$$

- $\sum_{k_0=0}^n \ldots \sum_{k_t=0}^n \|\Psi(k_0, \dots, k_t)\|^2 = 1$;
- $\dim(H(i, j)) \leq \dim(H_2)$ for all i, j , where $H(i, j)$ is the Hilbert space spanned by vectors $\{|\Psi^{t-i}(a_0, \dots, a_{i-1}, j)\rangle : a_k \in \mathbb{Z}_{n+1}\}$; and,
- $n = \dim(H_1) - 1$.

Theorem 2. *If an indexed set of vectors $\{|\Psi(k)\rangle \in H_A : k \in \mathbb{Z}_{n+1}^{t+1}\}$ is associated with an algorithm $\mathcal{A} = (t, n, m, H_Q, H_W, \Psi, \{U_i\})$, then it is a Block Set for (H_Q, H_W) .*

Proof. We divide the proof into four parts, each corresponding to one of the four properties from the definition of a Block Set:

1. Since $\{\tilde{P}_k^{t-i} : 0 \leq k \leq n\}$ is a CSOP, and using the fact that

$$|\Psi^i(a_0, a_1, \dots, a_{t-i})\rangle = \tilde{P}_{a_{t-i}}^{t-i} \dots \tilde{P}_{a_1}^1 \tilde{P}_{a_0}^0 |\Psi\rangle,$$

then, whenever $b_{t-i} \neq c_{t-i}$, we have $\langle \Psi^i(b_1, \dots, b_{t-i}) | \Psi^i(c_1, \dots, c_{t-i}) \rangle = 0$.

2. The second property is proved by using CSOP properties and mathematical induction.
3. The space generated by each $\{|\Psi^i(a_0, \dots, a_{t-i-1}, j)\rangle : a_k \in \mathbb{Z}_{n+1}\}$ is the same space generated by $\{\tilde{P}_j^{t-i} \tilde{P}_{a_{t-i-1}}^{t-i-1} \dots \tilde{P}_{a_0}^0 |\Psi\rangle : a_k \in \mathbb{Z}_{n+1}\}$, which is a subspace of the space generated by $\{\tilde{U}_{t-i}^\dagger |j\rangle |w\rangle : w \in H_W\}$, with dimension $\dim(H_W)$.
4. Finally, the fourth property follows directly from $\dim(H_Q) = n + 1$.

□

Theorem 3. Let $\{|\Psi(k)\rangle \in H_A : k \in \mathbb{Z}_{n+1}^{t+1}\}$ be a Block Set for (H_Q, H_W) , then it is associated with some $\mathcal{A} = (t, n, m, H_Q, H_W, \Psi, \{U_i\})$.

Proof. First, notice that t and n are trivially obtained from

$$\{|\Psi(k)\rangle \in H_Q \otimes H_W : k_i \in \mathbb{Z}_{n+1}\}$$

and m can be trivially obtained from H_W . We still have to obtain the other elements. For the initial state, we take

$$|\Psi\rangle = \sum_{k_0=0}^n \dots \sum_{k_t=0}^n |\Psi(k_0, \dots, k_t)\rangle.$$

Now, we must prove that $|\Psi\rangle$ is a unit vector. By using

$$|\Psi^i(a_0, \dots, a_{t-i})\rangle = \sum_{j=0}^n |\Psi^{i-1}(a_0, \dots, a_{t-i}, j)\rangle$$

as well as the first item of the Block Set definition, we get

$$\| |\Psi^i(a_0, \dots, a_{t-i})\rangle \|^2 = \sum_{j=0}^n \| |\Psi^{i-1}(a_0, \dots, a_{t-i}, j)\rangle \|^2.$$

Applying the previous equality recursively in $|\Psi\rangle$ and using the second item of the Block Set definition, we finally get

$$\| |\Psi\rangle \|^2 = \sum_{k_0=0}^n \dots \sum_{k_t=0}^n \| |\Psi(k_0, \dots, k_t)\rangle \|^2 = 1.$$

In the third part of the proof, we have to construct the unitary operators of \mathcal{A} . Those operators are obtained by using a construction of the CSOP sequence satisfying Eq. (5) with the Block Set $\{|\Psi(k)\rangle\}$. We define $H_1^i = \bigoplus_j H(i, j)$ and an orthogonal space H_2^i , such that $H_A = H_1^i \oplus H_2^i$. We have $\dim(H(i, j)) \leq \dim(H_W)$ from the third property of Definition 2. If $B(i)$ is an orthogonal basis of H_2^i , then for each pair (i, j) we can take $\dim(H_W) - \dim(H(i, j))$ linearly independent elements from $B(i)$ and write such set as B_j^i . The space generated by B_j^i is represented as $\widehat{H}(i, j)$. We also define a space $\widetilde{H}(i, j) = \widehat{H}(i, j) \oplus H(i, j)$ with the same dimension of H_W . We take the Hilbert spaces $\widehat{H}(i, j)$, which are pairwise orthogonal for different j . This is possible because $0 \leq j \leq n$ and $\dim(H_A) = (n+1) \dim(H_W)$. Thereby $j_1 \neq j_2$ implies that $\widetilde{H}(i, j_1)$ and $\widetilde{H}(i, j_2)$ are orthogonal. Then, for each i there is a CSOP $\{\widetilde{P}_j^i : 0 \leq j \leq n\}$ such that $\widetilde{H}(i, j)$ is the range of the projector \widetilde{P}_j^i . From Lemma 2, there is a unitary operator \widetilde{U}_i such that $\widetilde{U}_i^\dagger P_j \widetilde{U}_i = \widetilde{P}_j^i$, as the CSOP $\{P_k : 0 \leq k \leq n\}$ was defined. Thus, we obtain the unitary operators from $U_0 = \widetilde{U}_0$ and $U_i = \widetilde{U}_i \widetilde{U}_{i-1}^\dagger$ for $i > 0$. \square

Thus, we can say that for any algorithm there is a Block Set, and for any Block Set there is an algorithm. The reformulation is almost complete, except for one question: while an algorithm is associated to a unique Block Set, one Block Set may be associated to multiple algorithms. The following theorem implies that a non-bijective relation between both models is not a problem.

Theorem 4. *If two different algorithms are associated to the same Block Set*

$$\{|\Psi(k)\rangle \in H_A : k \in \mathbb{Z}_{n+1}^{t+1}\},$$

then they have the same Gram matrices for their final states.

Proof. As it was defined, a set $\{|\Psi(k)\rangle \in H_A : k \in \mathbb{Z}_{n+1}^{t+1}\}$ associated to an algorithm just depends on the unitary operators before the last query. Suppose that two algorithms are associated to the same set. From Corollary 1, we

have that the final state of the algorithms is equal to the same linear combination of elements from the Block Set for a fixed input x , but is different in the unitary operators applied over each sum. Then, $\langle \Psi_x^f | \Psi_y^f \rangle$ are equal in both algorithms. \square

Definition 3. *The output state of the input x under a Block Set*

$$\{|\Psi(k)\rangle \in H_A : k \in \mathbb{Z}_{n+1}^{t+1}\}$$

is defined as

$$|\Psi_x^f\rangle = \sum_{k_t=0}^n \dots \sum_{k_0=0}^n (-1)^{\sum_{i=0}^t x_{k_i}} |\Psi(k_0, \dots, k_t)\rangle. \quad (12)$$

This definition closes our new formulation, by defining the output states from the Block Set that describes the algorithm. Notice that the space H_A is maintained and the Gram matrix of final states from such Block Set is equal to the Gram matrix for final states of any algorithm associated to the Block Set. If we keep the same measurement step as in the original model, then we can compute the same functions within the same margin of error in the associated BSF, as we would with the QQM algorithms. In fact, it is just a matter of choosing adequate measurement steps. In Appendix A we present an example of a Block Set equivalent to a QQM algorithm.

4 Gram matrices and Block Sets

At this point, Block Sets are taken as an equivalent parametrization of quantum query algorithms, where we consider the elements of a Block Set as the new parameters. In this section, we study how each element will affect the final Gram matrix of output states. That information can open the possibility of using such parameters for constructing a Gram matrix, that is appropriate for computing a given function. If inputs x and y should give different outputs for a given function, then the quantum algorithm must be designed for making $\langle \Psi_x^f | \Psi_y^f \rangle$ as close to zero as possible.

It is convenient to introduce four auxiliary vectors, as follows. Vector $|A\rangle$ is defined as the sum of those components $|\Psi(a)\rangle$ of a Block Set whose sign is kept unchanged in both $|\Psi_x^f\rangle$ and $|\Psi_y^f\rangle$. Analogously, vector $|B\rangle$ is defined as the sum of those components $|\Psi(a)\rangle$ of a Block Set whose sign is kept unchanged

in $|\Psi_x^f\rangle$ while inverted in $|\Psi_y^f\rangle$. Vector $|C\rangle$ is defined as the sum of those components $|\Psi(a)\rangle$ of a Block Set whose sign is inverted in both $|\Psi_x^f\rangle$ and $|\Psi_y^f\rangle$. Finally, vector $|D\rangle$ is defined as the sum of those components $|\Psi(a)\rangle$ of a Block Set whose sign is inverted in $|\Psi_x^f\rangle$ while kept unchanged in $|\Psi_y^f\rangle$. Notice that $|\Psi\rangle = |A\rangle + |B\rangle + |C\rangle + |D\rangle$, and $|\Psi_x^f\rangle = |A\rangle + |B\rangle - |C\rangle - |D\rangle$, and $|\Psi_y^f\rangle = |A\rangle - |B\rangle - |C\rangle + |D\rangle$.

Expanding $\langle\Psi_x^f|\Psi_y^f\rangle$ and $\langle\Psi|\Psi\rangle$ in terms of the above defined vectors and summing those expressions, we get

$$\langle\Psi_x^f|\Psi_y^f\rangle = 2(\langle\Psi_x^+|\Psi_y^+\rangle + \langle\Psi_x^-|\Psi_y^-\rangle) - 1, \quad (13)$$

where $|\Psi_x^\pm\rangle = \frac{\pm|\Psi_x^f\rangle + |\Psi\rangle}{2}$, and analogously for $|\Psi_y^\pm\rangle$.

We say that a Block Set is real-valued if its elements are vectors on the real numbers.

Lemma 3. *If there is a complex Block Set $\{|\Psi(k)\rangle \in H_A : k \in \mathbb{Z}_{n+1}^{t+1}\}$ for (H_Q, H_W) , whose output states are used for computing a function $f : \{0, 1\}^n \rightarrow \{0, 1\}$ within error ϵ , then there is a real Block Set $\{|\hat{\Psi}(k)\rangle \in \hat{H}_Q \otimes \hat{H}_W : k \in \mathbb{Z}_{n+1}^{t+1}\}$ for some (\hat{H}_Q, \hat{H}_W) , whose output spaces can be used to compute f within the same error.*

Proof. If the outputs from $\{|\Psi(k)\rangle\}$ can be used for computing f within error ϵ (with an appropriate CSOP), then the existence of a quantum query algorithm that computes f within error ϵ in $t+1$ queries follows directly from Theorems 1 and 3. Barnum et al. [8] proved that there exists a quantum algorithm that computes f within error ϵ in $t+1$ queries, if and only if a semi-definite program $P(f, t+1, \epsilon)$ is feasible, where the unitary matrices and states in the quantum query algorithm corresponding to a solution for $P(f, t+1, \epsilon)$ can be taken to be real; Montanaro, Jozsa and Mitchison [5] gave an explicit construction achieving this. The set of vectors $\{|\hat{\Psi}(k)\rangle\}$ associated to this algorithm has output states that produces the same Gram matrix by Theorem 1 and all its elements are real. Finally, this set of vectors is a Block Set according to Theorem 2. \square

According to Lemma 3, we can always assume that Block Sets are real-valued, without loss of generality. The following lemma presents a useful property about this particular case.

Lemma 4. *If a Block Set $\{|\Psi(k)\rangle \in H_A : k \in \mathbb{Z}_{n+1}^{t+1}\}$ is real, then for any input $x \in \{0, 1\}^n$, $\langle\Psi_x^+|\Psi_x^-\rangle = 0$.*

Proof. There is $\langle \Psi_x^+ | \Psi_x^- \rangle = \frac{1}{4} \left(\|\Psi\|^2 + \langle \Psi_x^f | \Psi \rangle - \langle \Psi | \Psi_x^f \rangle - \|\Psi_x^f\|^2 \right)$.

If the Block Set is real, then $|\Psi\rangle$ and $|\Psi_x^f\rangle$ are real unit vectors, then $\langle \Psi_x^f | \Psi \rangle = \langle \Psi | \Psi_x^f \rangle$, in addition $\|\Psi\| = \|\Psi_x^f\| = 1$ implies $\langle \Psi_x^+ | \Psi_x^- \rangle = 0$. \square

Theorem 5. *Let the vectors $|A\rangle$, $|B\rangle$, $|C\rangle$, and $|D\rangle$ be as they were defined, however with the additional condition of being real-valued. Then we get*

$$\langle \Psi_x^f | \Psi_y^f \rangle = 2 \left(\|A\|^2 - 2 \langle A | C \rangle + \|C\|^2 \right) - 1. \quad (14)$$

Proof. Using Lemma 4 over $(|\Psi_x^+\rangle, |\Psi_x^-\rangle)$ and $(|\Psi_y^+\rangle, |\Psi_y^-\rangle)$, there are two equations. We consider a system of equations, joining the two last equations with Eq. (13). Expressing that system in dot products of $|A\rangle$, $|B\rangle$, $|C\rangle$ and $|D\rangle$, we obtain a new system that derives Eq. (14), by elementary algebra. \square

The previous theorem give us a way of obtaining the Gram matrix of final states directly from a given Block Set.

Let $\mathcal{B} = \{|\Psi(k)\rangle \in H_A : k \in \mathbb{Z}_{n+1}^{t+1}\}$ be a Block Set for (H_Q, H_W) . We denote

$$k = (k_0, k_1, \dots, k_t)$$

and the following subsets of \mathcal{B} :

1. $\mathcal{B}_x^+ = \{|\Psi(k)\rangle \in H_A : (-1)^{\sum_{i=0}^t x_{k_i}} = 1\}$.
2. $\mathcal{B}_x^- = \{|\Psi(k)\rangle \in H_A : (-1)^{\sum_{i=0}^t x_{k_i}} = -1\}$.

Then $\tilde{A}_{xy} = \mathcal{B}_x^+ \cap \mathcal{B}_y^+$ and $\tilde{C}_{xy} = \mathcal{B}_x^- \cap \mathcal{B}_y^-$.

Notice that \mathcal{B}_x^+ and \mathcal{B}_x^- are the sets of positive and negative terms in Eq. (9), respectively. So for each pair x, y , the sets \tilde{A}_{xy} and \tilde{C}_{xy} contain vectors of a Block Set, whose sum define $|A\rangle$ and $|C\rangle$, respectively.

Lemma 5. *Let $\mathcal{B} = \{|\Psi(k)\rangle \in H_A : k \in \mathbb{Z}_{n+1}^{t+1}\}$ be a Block Set for (H_Q, H_W) , where:*

- $P(k)$ is the set of pairs (x, y) such that $|\Psi(k)\rangle \in \tilde{A}_{xy}$.
- $Q(k)$ is the set of pairs (x, y) such that $|\Psi(k)\rangle \in \tilde{C}_{xy}$.

Then $P(k) = \{x : (x_{k_0} \oplus \dots \oplus x_{k_t}) = 0\}^2$ and $Q(k) = \{x : (x_{k_0} \oplus \dots \oplus x_{k_t}) = 1\}^2$.

Proof. Using the definitions of \tilde{A}_{xy} and \tilde{C}_{xy} , we have

$$P(k) = \left\{ (x, y) : (-1)^{\sum_{i=0}^t x_{k_i}} = 1 \text{ and } (-1)^{\sum_{i=0}^t y_{k_i}} = 1 \right\} \quad (15)$$

and

$$Q(k) = \left\{ (x, y) : (-1)^{\sum_{i=0}^t x_{k_i}} = -1 \text{ and } (-1)^{\sum_{i=0}^t y_{k_i}} = -1 \right\}. \quad (16)$$

Notice that x does not have influence on the predicate of y , nor y have influence on the predicate of x . Therefore, the sets of allowed values for x and y form a Cartesian product. Notice that $x_{k_0} \oplus \dots \oplus x_{k_t} = 0$ iff $(-1)^{\sum_{i=0}^t x_{k_i}} = 1$. \square

Now we may define the square matrices $\bar{P}_{k,h}$ and $\bar{Q}_{k,h}$, with row x and column y being indexed by elements of $\{0, 1\}^n$ and with entries taking values in $\{0, 1\}$, as follows:

- $\bar{P}_{k,h}[x, y] = 1$ iff $(x, y) \in P(k) \cap P(h)$;
- $\bar{Q}_{k,h}[x, y] = 1$ iff $(x, y) \in Q(k) \cap Q(h)$; and,
- $\bar{R}_{k,h}[x, y] = 1$ iff $(x, y) \in P(k) \cap Q(h)$.

Theorem 6. Let $\mathcal{B} = \{|\Psi(k)\rangle \in H_A : k \in \mathbb{Z}_{n+1}^{t+1}\}$ be a real Block Set for (H_Q, H_W) , then the Gram matrix of their output states $\{|\Psi_x^f\rangle\}$ is

$$G = 2 \sum_{k,h} (\bar{P}_{k,h} - 2\bar{R}_{k,h} + \bar{Q}_{k,h}) \langle \Psi(k) | \Psi(h) \rangle - J, \quad (17)$$

where J is a matrix where every element is equal to one.

Proof. Follows directly from Eq. (14), by rewriting the matrices $\{\bar{P}_{k,h}\}$, $\{\bar{R}_{k,h}\}$ and $\{\bar{Q}_{k,h}\}$. \square

This theorem gives an explicit expression on how pairs of elements in a Block Set control the Gram matrix of output states. We can think of each matrix $\bar{P}_{k,h} - 2\bar{R}_{k,h} + \bar{Q}_{k,h}$ like acting as a mask over the Gram matrix. Instead of this general case, there is a simpler case computationally less powerful, however with a simpler Gram matrix representation.

Definition 4. A Block Set $\mathcal{B} = \{|\Psi(k)\rangle \in H_A : k \in \mathbb{Z}_{n+1}^{t+1}\}$ for (H_Q, H_W) is orthogonal, if all its elements are orthogonal.

Corollary 2. Let $\mathcal{B} = \{|\Psi(k)\rangle \in H_A : k \in \mathbb{Z}_{n+1}^{t+1}\}$ be an orthogonal real Block Set for (H_Q, H_W) , then the Gram matrix of their output states $\{|\Psi_x^f\rangle\}$ is

$$G = 2 \sum_k (\bar{P}_{k,k} + \bar{Q}_{k,k}) \|\Psi(k)\|^2 - J. \quad (18)$$

Proof. Simply applying in Eq. (17) that $\langle \Psi(k) | \Psi(h) \rangle = 0$ for $k \neq h$ and $\bar{R}_{k,k} = 0$ for all k . \square

In Appendix B, we apply the ideas introduced in this section, and show explicitly how the Block Set determines the Gram matrix of output states through Theorem 6.

5 Towards a framework for analyzing quantum exact algorithms

In this section we introduce the BSF as tool for designing and analyzing exact quantum algorithms, this formulation implies linear systems that can admit analytic solutions. We also give examples of this application.

First, we define the set of unknowns $\{w_{kh} : k, h \in \mathbb{Z}_{n+1}^{t+1}\}$ for the set \mathbb{Z}_{n+1}^{t+1} . Let $X, Y \subset \{0, 1\}^n$ be two disjoint sets. From this notation, we may consider some useful equations:

1. For each $(x, y) \in X \times Y$ there is an equation

$$\sum_{k, h \in \mathbb{Z}_{n+1}^{t+1}} (\bar{P}_{k,h}[x, y] - 2\bar{R}_{k,h}[x, y] + \bar{Q}_{k,h}[x, y]) w_{kh} = \frac{1}{2}. \quad (19)$$

2. Let $\mathcal{I}_i(k') = \{k \in \mathbb{Z}_{n+1}^{t+1} : 0 \leq j \leq i \text{ and } k'_j = k_j \text{ for all } j\}$ for each $k' \in (\mathbb{Z}_{n+1})^{i+1}$. Thus, for each $i \in \mathbb{Z}_{t+1}$ and $k', h' \in (\mathbb{Z}_{n+1})^{i+1}$, such that $k'_i \neq h'_i$, there is an equation

$$\sum_{k \in \mathcal{I}_i(k')} \left(\sum_{h \in \mathcal{I}_i(h')} w_{kh} \right) = 0. \quad (20)$$

3. And, finally, there is a constraint

$$\sum_{k \in \mathbb{Z}_{n+1}^{t+1}} w_{kk} = 1. \quad (21)$$

The union of all these equations forms a system, which we denote as $E(t, n, X, Y)$.

Theorem 7. *Let $f : \{0, 1\}^n \rightarrow \{0, 1\}$ be a partial function such that, if $x \in X$ and $y \in Y$, then $f(x) \neq f(y)$. Then, f is computed exactly in $t + 1$ queries if and only if $E(t, n, X, Y)$ has a real solution for $\{w_{kh} : k, h \in \mathbb{Z}_{n+1}^{t+1}\}$ such that these values under the same indices form a positive semi-definite matrix.*

Proof. In the first part of the proof, if f can be computed exactly within $t + 1$ queries by a quantum query algorithm \mathcal{A} , then there is a set $\{|\Psi(k)\rangle : k \in \mathbb{Z}_{n+1}^{t+1}\}$ that is associated to the algorithm \mathcal{A} and this set is a Block Set according to Theorem 2. If this Block Set is complex then according to Lemma 3 there is another real Block Set $\{|\hat{\Psi}(k)\rangle : k \in \mathbb{Z}_{n+1}^{t+1}\}$, whose output states can be used for computing the same function f exactly.

Take $w_{k_1 k_2} = \langle \hat{\Psi}(k_1) | \hat{\Psi}(k_2) \rangle$. Since f is computed exactly, if $x \in X$ and $y \in Y$, then $f(x) \neq f(y)$ and the output states of \mathcal{A} must be orthogonal, i.e., $\langle \hat{\Psi}_x^f | \hat{\Psi}_y^f \rangle = 0$. Since \mathcal{A} and the Block Set have the same Gram matrix for output states, then from Theorem 6 we have that Eq. (19) is satisfied for (x, y) . Eq. (20) is just another way of writing the first property of Definition 2. Eq. (21) is another way of writing the second property of Definition 2. Finally, the values assigned for $\{w_{k_1 k_2}\}$ are a positive semi-definite matrix because it is the Gram matrix of $\{\hat{\Psi}(k)\}$.

In the second part of the proof, since the values for $\{w_{k_1 k_2}\}$ form a positive semi-definite matrix then it is a Gram matrix for a set of vectors $\{|\Psi(k)\rangle : k \in \mathbb{Z}_{n+1}^{t+1}\}$. This set of vectors satisfy the first property of Definition 2 according to Eq. (20) and the second property 2 of Definition 2 according to Eq. (21). If we define the appropriate spaces H_1 and H_2 , then the third and fourth properties of Definition 2 are satisfied and $\{\Psi(k)\}$ is a Block Set. From Eq. (19) and Theorem reffhema , we have that the sets of output states $\{|\Psi_x^f\rangle, x \in X\}$ and $\{|\Psi_y^f\rangle, y \in Y\}$ generate two orthogonal spaces. Therefore, there is a CSOP that allows us to measure the output exactly. From Theorems 1 and reffefff , we can conclude that a quantum query algorithm associated to $\{\Psi(k)\}$ jointly with the CSOP computes f exactly in $t + 1$ queries. \square

System $E(t, n, X, Y)$ has an exponential number of variables, then using this theorem for any numerical procedure is impractical and the theorem itself is difficult to use as an analytic tool. Another difficulty is maintaining the semi-definite property of the solution. Nevertheless there exists the possibility of taking special cases of this general formulation. For example, if we assume that

some variables are equal to zero, then we can construct particular families of exact quantum algorithms more easily. This is the strategy that we use in the following corollary for obtaining a more practical tool.

Let the system $\hat{E}(t, n, X, Y)$ be the union of the following equations:

$$\sum_{k \in \mathbb{Z}_{n+1}^{t+1}} (\bar{P}_{k,k}[x, y] + \bar{Q}_{k,k}[x, y]) w_{kk} = \frac{1}{2}, \quad (22)$$

for each $(x, y) \in X \times Y$, and

$$\sum_{k \in \mathbb{Z}_{n+1}^{t+1}} w_{kk} = 1. \quad (23)$$

Corollary 3. *Let $f : \{0, 1\}^n \rightarrow \{0, 1\}$ be a partial function where $x \in X$ and $y \in Y$ implies that $f(x) \neq f(y)$. If $\hat{E}(t, n, X, Y)$ has solution over the non-negative real numbers, then there is a quantum query algorithm that computes f exactly in $t + 1$ queries.*

Proof. If the Block Set has the restriction of being orthogonal (see Definition 4) for computing f , then taking $w_{k_1 k_2} = \langle \Psi(k_1) | \Psi(k_2) \rangle$ and $(k_1 \neq k_2)$ implies that $w_{k_1 k_2} = 0$. Then, Eq. (21) is the same as Eq. (23), Eq. (20) disappears and as $\bar{R}_{k,k} = 0$ then Eq. (19) becomes Eq. (22). Finally, the matrix formed by elements $w_{k_1 k_2}$ has no negative value in the diagonal and has zero in the rest, this guarantees the positive semi-definite property. \square

The orthogonality condition takes off computational power of the algorithms that we can obtain. However, this set of algorithms is still interesting. For example, it contains all exact quantum algorithms that use a single query. The largest possible separation between quantum and randomized query complexities can be obtained by a single query quantum algorithm—which is therefore orthogonal—even though this algorithm is not exact [1]. Corollary 3 is a much simpler tool, in the sense that each $k \in \mathbb{Z}_{n+1}^{t+1}$ has an independent influence to the Gram matrix. Let $T(k)$ be the set of pairs (x, y) such that

$$\bar{P}_{k,k}[x, y] + \bar{Q}_{k,k}[x, y] = 1.$$

We can say that the weight of $T(k)$ on the Gram matrix is controlled by the value of w_{kk} and the intersection of those sets determines which regions of $\{0, 1\}^n \times \{0, 1\}^n$ satisfy Eq. (22). That is equivalent to saying that those regions

have value 0 in the Gram matrix, and thus determines which inputs can be computed exactly for a given algorithm. However, the amount of weight that we can give to each k is limited by Eq. (23). It is also important to notice that increasing t increases the possible shapes for $T(k)$ and enlarges the set of possible Gram matrices that we can obtain. We can even imagine a random procedure for generating arbitrary exact quantum algorithms. The first step is giving weights for some set of variables $\{w_{kk} : k \in L \subset \mathbb{Z}_{n+1}^{t+1}\}$ until the limit imposed by Eq. (23) is reached, the last step is searching interesting sets X and Y such that $x \in X$ and $y \in Y$ iff

$$\sum_k (\bar{P}_{k,k}[x, y] + \bar{Q}_{k,k}[x, y]) w_{kk} = \frac{1}{2}.$$

The design of exact quantum algorithms using Corollary 3 can be done by analyzing the possible multiple intersections between the elements in set

$$\{T(k) : k \in \mathbb{Z}_{n+1}^{t+1}\}.$$

There are two useful observations that can be considered if we want to use Corollary 3. Let \bar{x} be the bit-wise negation of $x \in \{0, 1\}^n$, i.e., $\bar{x} \in \{0, 1\}^n$ such that $x_i \neq \bar{x}_i$ for all i . It is not difficult to prove that

$$\bar{P}_{k,k}[x, y] + \bar{Q}_{k,k}[x, y] = \bar{P}_{k,k}[\bar{x}, \bar{y}] + \bar{Q}_{k,k}[\bar{x}, \bar{y}]$$

for all k , and as a consequence all Gram matrix G obtained using the corollary have the property that $G[x, y] = G[\bar{x}, \bar{y}]$. Moreover, if $p(k)$ represents all the permutations of k , then $\bar{P}_{k,k} + \bar{Q}_{k,k} = \bar{P}_{k',k'} + \bar{Q}_{k',k'}$ for all $k' \in p(k)$. Thus, assigning random values to the set of unknowns $W(k) = \{w_{k'k'} : k' \in p(k)\}$, keeps the Gram matrix invariant as long as the sum $\sum_{k' \in W(k)} w_{k'k'}$ remains constant.

5.1 A generalization of the Deutsch-Jozsa algorithm by means of the Block Set Formalism

We show an example of BSF algorithm obtained by this analysis. We assume that n is even and $n > 2t$. Thereby, we define the set $\{k_i : 0 < i \leq n\} \subset \mathbb{Z}_{n+1}^{t+1}$, such that $k_i = (r(i), r(i+1), \dots, r(i+t))$, where $r(i) = i$ for $i \leq n$ and $r(i) = (i - n)$ for $i > n$. If we take $w_{k_i k_i} = \frac{1}{n}$ for all $0 < i \leq n$, then the system $\hat{E}(t, n, X, Y)$ is satisfied for $X = \{0^n, 1^n\}$ and $Y = \{x \in \{0, 1\}^n : S(x) = \frac{n}{2}\}$;

where we define $S(x)$ as the number of satisfied Boolean clauses $\phi_i = x_{r(i)} \oplus x_{r(i+1)} \oplus \dots \oplus x_{r(i+t)}$, such that $0 < i \leq n$. We can claim that $\widehat{E}(t, n, X, Y)$ is satisfied under the following observations. The equation

$$\sum_i (\bar{P}_{k_i, k_i} [0^n, y] + \bar{Q}_{k_i, k_i} [0^n, y]) w_{k_i k_i} = \frac{1}{2} \quad (24)$$

is satisfied only if $\frac{n}{2}$ matrices \bar{P}_{k_i, k_i} are equal to 1 in column y and row 0^n , because matrices \bar{Q}_{k_i, k_i} do not have values 1 on row 0^n . Last claims imply that $S(y) = \frac{n}{2}$. Finally, since $S(x) = \frac{n}{2}$ also implies that $S(\bar{x}) = S(x)$, we have that Eq. (24) must hold also for \bar{y} . Recall that $G[x, y] = G[\bar{x}, \bar{y}]$. Therefore,

$$\sum_i (\bar{P}_{k_i, k_i} [1^n, y] + \bar{Q}_{k_i, k_i} [1^n, y]) w_{k_i k_i} = \frac{1}{2} \quad (25)$$

for all y such that $S(y) = \frac{n}{2}$.

Thus, by Corollary 3 there is an exact quantum algorithm which computes two different outputs for X and Y . The first two cases of t are detailed below:

- For $t = 0$, there is a BSF algorithm equivalent to Deutsch-Jozsa algorithm [12].
- For $t = 1$, there is a BSF algorithm that discriminates $\{0^n, 1^n\}$ from x , where there is a set S such that $i \in S$ iff $x_i = x_{i+1}$ and $|S| = \frac{n}{2}$. This is stated by defining the first bit as following the last bit. This algorithm can be implemented in the QQM by applying Deutsch-Jozsa algorithm over the state $\sum_i (-1)^{x_i + x_j} |i\rangle$, where $j \equiv i + 1 \pmod n$, which costs two queries.

5.2 Characterizing the power of orthogonal algorithms

System $\widehat{E}(t, n, X, Y)$ implies a clear and straightforward view on how orthogonal BSF algorithms work, thus it is interesting in a theoretical sense. In practice however, we can work with a smaller system as it is proved below.

Theorem 8. *The system $\widehat{E}(t, n, X, Y)$ is equivalent to the system $\widetilde{E}(t, n, X, Y)$, which is defined as the union of following equations:*

$$\sum_{k \in \mathbb{Z}_{n+1}^{t+1}} \bar{P}_{k, k} [0^n, x \oplus y] w_{kk} = \frac{1}{2}. \quad (26)$$

for each $(x, y) \in X \times Y$, and

$$\sum_{k \in \mathbb{Z}_{n+1}^{t+1}} w_{kk} = 1. \quad (27)$$

Proof. Let $x \oplus y \in \{0, 1\}^n$ be the bit-wise xor operation between x and y . Consider the identity $\bar{P}_{k,k}[x, y] + \bar{Q}_{k,k}[x, y] = \bar{P}_{k,k}[0^n, x \oplus y]$. Then

$$\sum_{k \in \mathbb{Z}_{n+1}^{t+1}} (\bar{P}_{k,k}[x, y] + \bar{Q}_{k,k}[x, y]) w_{kk} = \sum_{k \in \mathbb{Z}_{n+1}^{t+1}} \bar{P}_{k,k}[0^n, x \oplus y] w_{kk}. \quad (28)$$

□

Last theorem implies that system $\tilde{E}(t, n, X, Y)$ is equivalent to $\hat{E}(t, n, 0^n, Z)$, where there is defined $Z = \{x \oplus y : (x, y) \in X \times Y\}$. Thereby, if an exact orthogonal BSF algorithm discriminates 0^n from Z , then it also can be used for discriminating X from Y with error zero. In the case of orthogonal BSF algorithms, Theorem 9 allows us to simplify the algorithm-construction problem, we just need to determine which sets can be discriminated from 0^n given a bounded t . Recall that a permutation on vector k gives the same variable w_{kk} , besides repeated values $k_i = k_j$ in k also implies redundancy, then system $\tilde{E}(t, n, 0^n, Z)$ implies a matrix of size $\mathcal{O}(2^n) \times \mathcal{O}(2^n)$. We can compare it with the system given by Barnum, Saks and Szegedy [8] which implies $\mathcal{O}(t)$ matrices of size $\mathcal{O}(2^n)$, thus $\tilde{E}(t, n, 0^n, Z)$ is less powerful but also computationally cheaper.

Corollary 4 characterizes the computational power of orthogonal exact algorithms, but first we define a problem that is general enough for describing any function whose domain is in the hypercube.

Definition 5 (XOR-Weighted-Problem). *Let be a set of Boolean formulas*

$$\mathbb{X} = \left\{ \bigoplus_i x_{k_i} : x_0 = 0, k \in K \subset \mathbb{Z}_{n+1}^{t+1} \right\},$$

where each formula is associated to a weight $w_{kk} > 0$ such that $\sum_{k \in K} w_{kk} = 1$. Consider m disjoint sets $X_i \subset \{0, 1\}^n$ and $Z = \{x \oplus y : (x, y) \in X_i \times X_j\}$, such that $z \in Z$ implies that $S(z)_w = \frac{1}{2}$, where $S(z)_w$ is the sum of weights of each formula in \mathbb{X} that is satisfied by z . The XOR-Weighted-Problem consists in separating sets X_i in different outputs.

Corollary 4. *Quantum exact algorithms can solve the XOR-Weighted-Problem within $t + 1$ queries.*

Proof. This is a reformulation of Theorem 8. Notice that $x_0 = 0$ in \mathbb{X} is a consequence of Eq. (3). \square

Corollary 4 characterizes the power of orthogonal exact algorithms and it represents a model whose complexity upper-bounds the quantum query model. Fig. 2 gives a visualization of the XOR-Weighted-Problem.

6 A lower bound for exact quantum algorithms

In this section, using the BSF approach, we develop a lower bound result for exact quantum query complexity, considering functions of Boolean domain but arbitrary output.

We apply a basis for the Boolean cube [11], which is a family of functions

$$\mathcal{F}_k^n : \{0, 1\}^n \rightarrow \{1, -1\},$$

such that each $\mathcal{F}_k^n(x) = \prod_{i=0}^{n-1} (-1)^{x_{k_i}}$ is defined for vectors $k \in \mathbb{Z}_n^n$.

Consider that, for $k \neq h$, it is possible that $\mathcal{F}_k^n = \mathcal{F}_h^n$. Thereby, we need to define an equivalence relation $k \sim h$, for $k, h \in \mathbb{Z}_n^n$ such that $\mathcal{F}_k^n = \mathcal{F}_h^n$. We define \mathbb{Q}_n as the quotient set of our relation and the set $[k] \in \mathbb{Q}_n$ as the equivalence class for element k . We also define (a) \mathbb{F}_n , which elements are defined as functions indexed by \mathbb{Q}_n such that $\mathcal{F}_{[h]}^n = \mathcal{F}_k^n$ iff $k \in [h]$, and (b) $\mathbb{F}_n(m) \subset \mathbb{F}_n$, where $\mathcal{F}_{[k]} \in \mathbb{F}_n(m)$ iff $[k]$ contains an element h with no more than $2m$ non-zero terms. Finally, we define $\mathcal{F}_{[k,h]}^n : \{0, 1\}^n \rightarrow \{0, 1\}$, with output 1 iff $\mathcal{F}_{[k]}^n(x) = \mathcal{F}_{[h]}^n(x) = 1$. Notice that

$$\mathcal{F}_{[k,h]}^n(x) = \frac{\mathcal{F}_{[k]}^n(x) + \mathcal{F}_{[h]}^n(x) + \mathcal{F}_{[k \circ h]}^n(x) + 1}{4},$$

where $[k \circ h] \in \mathbb{Q}_n$ is an equivalence class such that $\mathcal{F}_{[k \circ h]}^n(x) = 1$ iff $\mathcal{F}_{[k]}^n(x) = \mathcal{F}_{[h]}^n(x)$.

For the following result we introduce additional notation. Let $w : \{0, 1\}^n \rightarrow \mathbb{R}$ be a function, and define $w * \mathcal{F}_{[h]}^n = \sum_{x \in \{0, 1\}^n} w(x) \mathcal{F}_{[h]}^n(x)$. We denote \bar{a} as a vector such that all its terms are a . Finally, let $\rho(i) = 0$ if i is even, and $\rho(i) = 1$ otherwise.

Theorem 9. Consider m disjoint sets $X_i \subset \{0, 1\}^n$, such that for each $x \in X_i$ there is a set $Z(x) = \{x \oplus y : y \in X_j \text{ and } j \neq i\}$. We also define a family of functions $g_y^k(x)$ such that (a) $g_y^k(\bar{0}) = 1$, (b) $g_y^k(x) = \frac{1}{2}$ for $x \in Z(y)$, (c) $g_y^k(x) = 1$ for x , where

$$\sum_{i=0}^{2k} \sum_{j=0} \binom{|x|}{i - 2j - \rho(i)} \binom{n - |x|}{2j + \rho(i)} > \frac{\sum_{i=0}^{2k} \binom{n}{i}}{2} \quad (29)$$

and (d) $g_y^k(x) = 0$ otherwise. If $\sum_{\mathcal{F}_{[h]}^n \in \mathbb{F}_n(k)} g_y^k * \mathcal{F}_{[h]}^n \geq 1$ for all $y \in \bigcup_i X_i$, then an exact quantum algorithm that gives different outputs for each X_i , applies at least k queries.

Proof. Suppose that a quantum algorithm allows us to separate $x \in X_i$ from $\bigcup_{j \neq i} X_j$, by applying k queries and without error. Using Gram matrix representation from Theorem 6 at row x , we have

$$\frac{1}{2} (G_{[x, x \oplus y]} + 1) = \sum_{[h] \in \mathbb{Q}_n} \alpha_{[h]} \mathcal{F}_{[h]}^n(y) + \sum_{[h_i] \neq [h_j]} \alpha_{[h_i, h_j]} \mathcal{F}_{[h_i, h_j]}^n(y) \quad (30)$$

Defining $\bar{T}_{h_1, h_2} = \bar{P}_{h_1, h_2} - 2\bar{R}_{h_1, h_2} + \bar{Q}_{h_1, h_2}$ from the matrices in Eq. (17), notice that first sum in the expression comes from \bar{T}_{h_1, h_2} when $h_1 = h_2$ and second sum comes from \bar{T}_{h_1, h_2} when $h_1 \neq h_2$. Thus, we have $\sum_{[h] \in \mathbb{Q}_n} \alpha_{[h]} = 1$ and

$\sum_{[h_i] \neq [h_j]} \alpha_{[h_i, h_j]} = 0$, that implies

$$\sum_{\mathcal{F}_{[h]}^n \in \mathbb{F}_n(k)} \frac{1}{2} (G_{[x, x \oplus y]} + 1) * \mathcal{F}_{[h]}^n = \sqrt{2^n}. \quad (31)$$

Thereby, we can state a necessary condition for an orthogonality between the final states of $x \in X_i$ and $\bigcup_{j \neq i} X_j$, where the algorithm applies k queries. That is the existence of some function $g : \{0, 1\}^n \rightarrow [0, 1]$, such that $g(x) = \frac{1}{2}$ for $x \in \bigcup_{j \neq i} X_j$, $g(\bar{0}) = 1$ and $\sum_{\mathcal{F}_{[h]}^n \in \mathbb{F}_n(k)} g * \mathcal{F}_{[h]}^n \geq \sqrt{2^n}$. The function $g_y^k(x)$ fulfills such properties maximizing $\sum_{\mathcal{F}_{[h]}^n \in \mathbb{F}_n(k)} g * \mathcal{F}_{[h]}^n$. That is, if $x \notin \bigcup_{j \neq i} X_j - \{\bar{0}\}$ then $g_y^k(x) = 1$ for inputs x such that there are more functions in $\mathbb{F}_n(k)$ with value 1

than -1 and $g_y^k(x) = 0$ otherwise. Notice that $\sum_{i=0}^{2k} \binom{n}{i}$ is the cardinality of $\mathbb{F}_n(k)$ and

$$\sum_{i=0}^{2k} \sum_{j=0}^{\lfloor \frac{n-i}{2} \rfloor} \binom{|x|}{i-2j-\rho(i)} \binom{n-|x|}{2j+\rho(i)}$$

is the cardinality of functions in $\mathbb{F}_n(k)$ with value 1 in x . \square

This theorem offers an alternative lower-bound to traditional tools like Polynomial and Adversary methods. We present a simple example of its application, that is the total function f that separates $X_1 = \{\bar{0}, \bar{1}\}$ from $X_2 = \{0, 1\}^n - X_1$. For any $y \in X_1$, we have $g_y^k(x) = \hat{g}_y(x) + \tilde{g}_y(x)$, where (a) $\hat{g}_y(\bar{0}) = 1$ and $\hat{g}_y(x) = \frac{1}{2}$ for $x \neq \bar{0}$, and (b) $\tilde{g}_y(\bar{1}) = \frac{1}{2}$ and $\tilde{g}_y(x) = 0$ for $x \neq \bar{1}$. That is because $k \leq \lfloor \frac{n}{2} \rfloor$ implies that Eq. (29) for $x = \bar{1}$ becomes

$$\sum_{i=0}^k \binom{n}{2i} > \frac{\sum_{i=0}^{2k} \binom{n}{i}}{2}.$$

Thus, we have

$$\sum_{\mathcal{F}_{[h]}^n \in \mathbb{F}_n(k)} \hat{g}_y * \mathcal{F}_{[h]}^n = \frac{1}{\sqrt{2^n}} \sum_{i=0}^{2k} \binom{n}{i}$$

and

$$\sum_{\mathcal{F}_{[h]}^n \in \mathbb{F}_n(k)} \tilde{g}_y * \mathcal{F}_{[h]}^n = \frac{1}{2\sqrt{2^n}} \sum_{i=0}^k \binom{n}{2i}.$$

Choosing $k = \lceil \frac{4n}{10} \rceil$, we have that

$$\sum_{i=0}^{\lceil \frac{4n}{10} \rceil} \binom{n}{2i} > 4 \left[\sum_{i=2\lceil \frac{4n}{10} \rceil}^n \binom{n}{i} \right].$$

That is enough for proving

$$\sum_{\mathcal{F}_{[h]}^n \in \mathbb{F}_n(\lceil \frac{4n}{10} \rceil)} g_y^{\lceil \frac{4n}{10} \rceil} * \mathcal{F}_{[h]}^n \geq \sqrt{2^n},$$

which gives a lower bound $Q_E(f) = \Omega(n)$.

7 Conclusion

In this work, we presented tree theoretical results. Our main theoretical result was the Block Set Formulation, which is a reformulation of the Quantum Query Model such that the unitary operators are replaced by phase inversions over a set of vectors. This contribution gives an alternative interpretation on how quantum query algorithms work. A second result is a linear system of equations that allows an alternative analysis and construction of quantum exact algorithms for partial functions. These constructions are delimited by a problem defined by weights over formulas, which can be considered a model that upper-bounds the QQM. Finally, we apply the BSF approach for developing a lower-bound for exact quantum algorithms. These results give a validation of our formulation.

This approach leaves open problems and research possibilities:

- It is possible to obtain algorithms with some error by using the introduced tools, for example by approximate solutions to system $E(t, n, X, Y)$, but this does not guarantee a bounded error. This approach would be extended by finding a sufficient and necessary condition for obtaining a bounded error algorithm.
- The condition $(k_1 \neq k_2 \Rightarrow w_{k_1 k_2} = 0)$ used in Corollary 3 could be weakened for some unknowns obtaining more powerful yet complicated models than the orthogonal BSF. Which strategies can be developed for constructing exact quantum algorithms under the general BSF (Theorem 7) or a weaker condition?

Acknowledgements

This work received financial support from CAPES and CNPq. The authors thank the group of Quantum Computing at LNCC/MCTI and the Laboratory of Algorithms and Combinatorics at PESC/COPPE/UFRJ for helpful discussions.

References

- [1] Scott Aaronson and Andris Ambainis. Forrelation: A problem that optimally separates quantum from classical computing. In *Proceedings of the Forty-Seventh Annual ACM on Symposium on Theory of Computing*, pages 307–316. ACM, 2015.

- [2] Andris Ambainis. Quantum lower bounds by quantum arguments. In *Proceedings of the thirty-second annual ACM symposium on Theory of computing*, pages 636–643. ACM, 2000.
- [3] Andris Ambainis. Superlinear advantage for exact quantum algorithms. *SIAM Journal on Computing*, 45(2):617–631, 2016.
- [4] Andris Ambainis, Kaspars Balodis, Aleksandrs Belovs, Troy Lee, Miklos Santha, and Juris Smotrovs. Separations in query complexity based on pointer functions. arXiv preprint arXiv:1506.04719, 2015.
- [5] Andris Ambainis, Jozef Gruska, and Shenggen Zheng. Exact quantum algorithms have advantage for almost all boolean functions. arXiv preprint arXiv:1404.1684, 2014.
- [6] Andris Ambainis, Jānis Iraids, and Daniel Nagaj. Exact quantum query complexity of $\text{EXACT}_{k,1}^n$. arXiv preprint arXiv:1608.02374, 2016.
- [7] Andris Ambainis, Jānis Iraids, and Juris Smotrovs. Exact quantum query complexity of exact and threshold. arXiv preprint arXiv:1302.1235, 2013.
- [8] Howard Barnum, Michael Saks, and Mario Szegedy. Quantum decision trees and semidefinite. Technical Report LA-UR-01-6417, Los Alamos National Laboratory, May 2002.
- [9] Robert Beals, Harry Buhrman, Richard Cleve, Michele Mosca, and Ronald De Wolf. Quantum lower bounds by polynomials. *Journal of the ACM (JACM)*, 48(4):778–797, 2001.
- [10] H. Buhrman and R. de Wolf. Complexity measures and decision tree complexity: A survey. *Theoretical Computer Science*, 288(1):21–43, October 1999.
- [11] R. De Wolf. A brief introduction to fourier analysis on the boolean cube. *Theory of Computing*, Graduate Surveys, 2008.
- [12] David Deutsch and Richard Jozsa. Rapid solution of problems by quantum computation. In *Proceedings of the Royal Society of London A: Mathematical, Physical and Engineering Sciences*, volume 439, pages 553–558. The Royal Society, 1992.

- [13] Jozef Gruska, Daowen Qiu, and Shenggen Zheng. Generalizations of the distributed deutsch–jozsa promise problem. *Mathematical Structures in Computer Science*, pages 1–21, 2015.
- [14] Peter Hoyer, Troy Lee, and Robert Spalek. Negative weights make adversaries stronger. In *Proceedings of the thirty-ninth annual ACM symposium on Theory of computing*, pages 526–535. ACM, 2007.
- [15] Phillip Kaye, Raymond Laflamme, and Michele Mosca. *An introduction to quantum computing*. Oxford University Press, 2007.
- [16] Gatis Midrijanis. Exact quantum query complexity for total boolean functions. arXiv preprint quant-ph/0403168, 2004.
- [17] Ashley Montanaro, Richard Jozsa, and Graeme Mitchison. On exact quantum query complexity. *Algorithmica*, 71(4):775–796, 2015.

Appendix A

In this appendix, we give a simple one-dimensional example of Block Set associated to a QQM algorithm, namely Deutsch’s algorithm. For simplicity our H_W is an empty set, thus the algorithm has an initial state $\Psi_0 = |0\rangle$. The QQM representation of Deutsch’s algorithm takes the unitary operators

$$U_0 = \begin{bmatrix} 0 & 0 & 1 \\ \frac{1}{\sqrt{2}} & \frac{1}{\sqrt{2}} & 0 \\ \frac{1}{\sqrt{2}} & -\frac{1}{\sqrt{2}} & 0 \end{bmatrix},$$

and $U_1 = I$. The CSOP used by the measurement step is not important for our purposes. Considering the CSOP $\{P_k\}$ as defined in Section 3, we have $P_i = |i\rangle\langle i|$. Using Definition 1, we obtain each element of the Block Set, namely

$$|\Psi(0)\rangle = \tilde{P}_0^0 |0\rangle = U_0^\dagger P_0 U_0 |0\rangle = \begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix},$$

$$|\Psi(1)\rangle = \tilde{P}_1^0 |0\rangle = U_0^\dagger P_1 U_0 |0\rangle = \begin{bmatrix} \frac{1}{2} \\ \frac{1}{2} \\ 0 \end{bmatrix},$$

and

$$|\Psi(2)\rangle = \tilde{P}_2^0 |0\rangle = U_0^\dagger P_2 U_0 |0\rangle = \begin{bmatrix} \frac{1}{2} \\ -\frac{1}{2} \\ 0 \end{bmatrix}.$$

Take $\{|\Psi_x^f\rangle\}$ and $\{|\tilde{\Psi}_x^f\rangle\}$ as the final states of Deutsch's algorithm and the Block Set, respectively. We consider that $x = x_n \dots x_2 x_1$. Using Definition 3 and Theorem 1, we have

$$|\tilde{\Psi}_{00}^f\rangle = |\Psi(0)\rangle + |\Psi(1)\rangle + |\Psi(2)\rangle = \begin{bmatrix} 1 \\ 0 \\ 0 \end{bmatrix},$$

$$|\tilde{\Psi}_{01}^f\rangle = |\Psi(0)\rangle - |\Psi(1)\rangle + |\Psi(2)\rangle = \begin{bmatrix} 0 \\ -1 \\ 0 \end{bmatrix},$$

$$|\tilde{\Psi}_{10}^f\rangle = |\Psi(0)\rangle + |\Psi(1)\rangle - |\Psi(2)\rangle = \begin{bmatrix} 0 \\ 1 \\ 0 \end{bmatrix}$$

and

$$|\tilde{\Psi}_{11}^f\rangle = |\Psi(0)\rangle - |\Psi(1)\rangle - |\Psi(2)\rangle = \begin{bmatrix} -1 \\ 0 \\ 0 \end{bmatrix}.$$

Since in this case we have the identities

$$|\Psi_{00}^f\rangle = \begin{bmatrix} 0 \\ \frac{1}{\sqrt{2}} \\ \frac{1}{\sqrt{2}} \end{bmatrix} = -|\Psi_{11}^f\rangle$$

and

$$|\Psi_{01}^f\rangle = \begin{bmatrix} 0 \\ -\frac{1}{\sqrt{2}} \\ \frac{1}{\sqrt{2}} \end{bmatrix} = -|\Psi_{10}^f\rangle,$$

if we calculate the Gram matrices of $\{|\Psi_x^f\rangle\}$ and $\{|\tilde{\Psi}_x^f\rangle\}$, then we obtain the

same matrix

$$G = \begin{bmatrix} 1 & 0 & 0 & -1 \\ 0 & 1 & -1 & 0 \\ 0 & -1 & 1 & 0 \\ -1 & 0 & 0 & 1 \end{bmatrix}.$$

In both algorithms, the final states for inputs $X = \{00, 11\}$ are orthogonal to the final states for inputs $Y = \{01, 10\}$. Thereby there exist CSOPs that discriminate X from Y within error 0, for both algorithms. This example show that both QQM and BSF algorithms are equivalent to Deutsch's algorithm by choosing the appropriate measurement steps.

Appendix B

Here, we extend our previous example of Block Set obtained from Deutsch's algorithm. This extension shows concepts introduced by Section 4. In our example, all one-dimensional Block Sets are orthogonal, thus this algorithm is represented by Corollary 2. In other words, for this case, $k \neq h \Rightarrow \bar{P}_{k,h} + \bar{Q}_{k,h} = 0$. Thereby, we are just interested in matrices of the form $\bar{P}_{k,k} + \bar{Q}_{k,k}$. Matrices for each element k are given by

$$\bar{P}_{0,0} + \bar{Q}_{0,0} = \begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 \end{bmatrix},$$

$$\bar{P}_{1,1} + \bar{Q}_{1,1} = \begin{bmatrix} 1 & 0 & 1 & 0 \\ 0 & 1 & 0 & 1 \\ 1 & 0 & 1 & 0 \\ 0 & 1 & 0 & 1 \end{bmatrix}$$

and

$$\bar{P}_{2,2} + \bar{Q}_{2,2} = \begin{bmatrix} 1 & 1 & 0 & 0 \\ 1 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 \\ 0 & 0 & 1 & 1 \end{bmatrix}.$$

Thus, calculating a matrix M with the Block Set obtained in Appendix A, we get

$$M = \sum_{i=0}^2 (\overline{P}_{i,i} + \overline{Q}_{i,i}) \langle \Psi(i) | \Psi(i) \rangle = \begin{bmatrix} 1 & \frac{1}{2} & \frac{1}{2} & 0 \\ \frac{1}{2} & 1 & 0 & \frac{1}{2} \\ \frac{1}{2} & 0 & 1 & \frac{1}{2} \\ 0 & \frac{1}{2} & \frac{1}{2} & 1 \end{bmatrix}.$$

We finally obtain the Gram matrix of the BSF algorithm from Corollary 2. Notice that the resulting matrix is the same as the obtained in Appendix A,

$$G = 2(M) - \begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & -1 \\ 0 & 1 & -1 & 0 \\ 0 & -1 & 1 & 0 \\ -1 & 0 & 0 & 1 \end{bmatrix}.$$

This showed how each element of the block set works as a parameter for the Gram matrix of final states. Thus, Eq. (18) is satisfied.

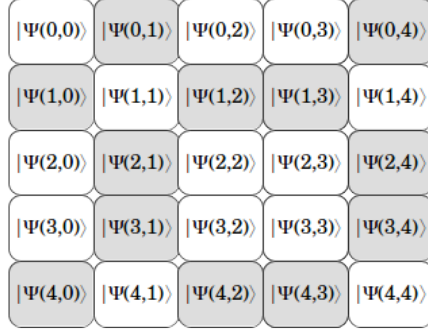


Figure 1: Graphical representation of $\tilde{P}_x^1 \tilde{P}_x^0 |\Psi\rangle$ using Eq. (9) and taking as input $x = 1001$. Grey boxes represent components where the relative phase is inverted with respect to the initial state $|\Psi\rangle$.

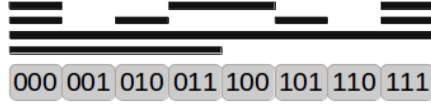


Figure 2: Each black layer represents the satisfiability of some formula over an input x . In decreasing order, the formulas are $x_1 \oplus x_2$, $x_1 \oplus x_3$, x_0 and x_3 . If we give the same weight $\frac{1}{4}$ to all these formulas, then any input x having exactly two layers over itself is orthogonal to 000. In our example 001, 100 and 101.